Procedia Computer Science

Volume 71, 2015, Pages 133–142

2015 Annual International Conference on Biologically Inspired Cognitive Architectures

A Parameter Estimation Method for Dynamic Computational Cognitive Models

Dilhan J. Thilakarathne

Agent Systems Research Group, VU University Amsterdam, The Netherlands
d.j.thilakarathne@vu.nl

Abstract
A dynamic computational cognitive model can be used to explore a selected complex cognitive phenomenon by providing some features or patterns over time. More specifically, it can be used to simulate, analyse and explain the behaviour of such a cognitive phenomenon. It generates output data in the form of time series which can only be partially compared to empirical knowledge. This leads to a challenging problem to estimate values of the parameters of the model representing characteristics of a person. A parameter estimation approach for dynamic cognitive models is presented here by combining improved Particle Swarm Optimization (PSO) and Constraint Satisfaction (CS) methods. Having collected the key features of behaviour of a phenomenon, those are translated into a set of constraints with parameters that will be solved through an improved agent based PSO technique. Through this, within PSO each agent explores the complex search space while communicating the quality of a local parameter value vector relative to their current global best solution as a swarm (through cooperation and competition). This is performed in tournaments and results of each tournament are combined to address the premature convergence issue in PSO.

Keywords: Parameter estimation, Particle swarm optimization, Constrain satisfaction, Cognitive modelling

1 Introduction

With the developments in brain-imaging and recording techniques, more and more detailed information on various brain processes becomes available and this contributes a strong increase in the development in cognitive modelling [1]. It has been found that more than 80% of published articles in theoretical journals in cognitive science are about cognitive modelling [2]. Furthermore, computational modelling is considered to be an important pillar for the development of cognitive science and its related disciplines [3]–[5]. In principle, in cognitive modelling, a phenomenon of human cognition (or behaviour) is represented as (computational) mathematical models to understand the causality with exposed adjustable parameters that can be estimated from empirical data to align with reality [2], [6]. The analogy between the actual human cognition and empirical evidences collecting through brain imaging and recording techniques occurs at two different levels of

Selection and peer-review under responsibility of the Scientific Programme Committee of BICA 2015

doi:10.1016/j.procs.2015.12.178

abstraction. Pragmatically, this nature of observed output (in most cases these will be in the form of features or patterns over time that explain the characteristics in non-quantitative, discrete forms) is very useful to understand the workings of the highly complex human brain. However, this makes it a non-trivial challenge to validate the behaviour of such a model, due to not having an exact quantitative outcome about features or patterns of human cognition. Almost all models developed in science and engineering include parameters (representing characteristics of a person or process) and that is a key driving element to bridge the gap between what is predicted and what is observed [7]. The most generic approach in parameter estimation is systematically changing the parameter values such that the error between predicted and observed comes closer to zero. This strategy is difficult to directly adopt for cognitive models and especially for dynamic cognitive models (see [6]).

Given the nature of available empirical data, first it is essential to translate these (incomplete) output data into a more well defined and quantifiable form. One of the most promising generic representations for this is to express those data as (temporal) constraints. Dynamic features and/or patterns can be easily represented as set of constraints using formal languages. Having a set of such constraints, the parameter estimation process condenses to a problem of Constraint Satisfaction (CS). A large number of problems in computing domains are represented directly or indirectly as Constraint Satisfaction Problems (CSP) [8], [9] and most of these problems belong to the class of NP-complete problems, including most of cognition related problems (see [10], [11]). Therefore, it is a challenge to develop a technique to solve a problem of constraint satisfaction that is capable of finding a solution within a reasonable computational expense. Stochastic and heuristic based searching techniques are commonly used for this purpose and each having specific advantages and disadvantages. Particle swarm optimization (PSO) technique is attracting the attention of many researchers in many domains due to its efficiency and effectiveness. Main advantages of the PSO methodology are simplicity in calculation, easiness in implementation, comparatively little number of parameters, and free from derivatives [12]. In this paper, a generic approach for parameter estimation in dynamic computational cognitive models is presented by combining CS and improved PSO techniques. It is illustrated for a complex cognitive model addressing action awareness.

2 Theoretical basis of CS and PSO

CS and PSO are well known techniques in artificial intelligence (PSO is relatively new) for many problems and both are having well defined mathematical basis. A constraint satisfaction problem is defined by a finite set of variables $X = \{x_1, x_2, ..., x_n\}$ (each variable x_i has its own domain D_{xi} that includes values $\{v_1, v_2, ..., v_m\}$) and a set of constraints $C = \{c_1, c_2, ..., c_p\}$ (each constraint C_i involves some subset of variables). The problem is said to be solved if it is possible to find an assignment of values to variables such that it satisfies restrictions imposed by all the constraints [9]. Even though the representation of a CSP is very simple it is a non-trivial task to find a solution tuple depend on the complexity of a given problem. Constraints restrict the values that each variable can simultaneously holds and the challenge is to identify a searching technique to find a global assignment to all the variables that satisfies all the constraints with both efficiently and reasonably less computational expense (for such techniques see [8], [9]).

PSO is a relatively new population based hybrid (both heuristic and stochastic) search/optimization technique that can be used to find (approximate) solution(s) in effective and efficient manner. PSO is first introduced by Kennedy and Eberhart [13], and it emulates a social system that can be observed in a flock of animals that have no leaders. These flocks achieve their goals through communication with others by sharing information about current situation, and therefore group will condense to a position which has a better solution. This phenomenon continues until the best conditions or a situation is discovered. PSO also consists with swarm of particles where each particle holds a position that represents a solution (in a search space) and has a velocity. Particles are moved based on three

configurable factors: social, cognitive, and speed [12], [14], [15]. Social factor influences the convergence towards the best solution discovered so far by a particle of the swarm, and cognitive factor influences each particle on its best position discovered so far, and speed factor delimit the movement. Therefore, always a movement is a resultant of these three factors. These three factors again coupled with another parameter set such that convergence can be biased to social, cognitive, speed or any permutation in these three (see [14]). This behaviour can be represented in two simple equations:

$$v_i(t+1) = \omega v_i(t) + c_1 r_1(t)\big(P_i - X_i(t)\big) + c_2 r_2(t)\left(P_g - X_i(t)\right) \quad (1)$$

$$X_i(t+1) = X_i(t) + v_i(t+1) \quad (2)$$

Where $v_i(t)$ represents velocity of the ith particle, P_i represents the best previous position of the i^{th} particle, P_g represents the best position discovered so far among all the particles in the swarm, $X_i(t)$ represents the position of the ith particle $X_i(t)= (x_{i1},x_{i2},...,x_{iD})$ (where D is the dimension of the search space), c_1 and c_2 are positive constants called acceleration coefficients (c_1: cognitive factor of the particles, c_2: Social factor of the particles), r_1 and r_2 are two random numbers in the range [0,1], and ω is the weight of inertia for the velocity.

In each iteration, each particle evaluates the quality of its current position. This evaluation can be combined with the CSP. In PSO, a position of a particle is a (probable) solution for the finite set of variables X mentioned in the CSP. Each particle shares its information with the swarm and updates the new position as in the equations (1) and (2). PSO has a major limitation due its premature convergence problem [15], [16]. As there is a biasness towards to the global best solution, having discovered a local optimum all agents will scatted towards to that. This can be handled by minimizing the influence towards the global best solution by providing a very small c_1 value but a large c_2 value (see [15]). Then the particles are having a problem of getting converge and they just explore around their local best position than exploring the search space proactively. Therefore, this premature convergence is not easy to handle and various approaches were presented in the literature (see [16]). For this paper, an improved PSO algorithm is scrutinized by considering the concept of tournament based selection. In each tournament, particles are trying to find a solution that satisfies all the constraints but in most cases, they may trap in a local optimum. If it is identified that the swarm is taped in a local optimum (with the same global best solution for 'm' number of consecutive iterations) the global best solution of that tournament stores and initiates a new tournament. After 'n' number of consecutive tournaments, swarm has identified n best local optimums (in the meanwhile, there is a possibility of finding the best solution and in such a case process is terminated). Having n number of local optimums, each particle is replaced with a local optimum and let swarm to explore the search space. This process continues until finding a global solution that satisfies all the constraints. Pseudocode of this improved PSO is provided in Pseudocode 1. There are some rule of thumbs to select parameter values for the PSO algorithm [12], [15]. Number of particles (n) to be selected is normally in the range of [20, 50], ω is advised to be in the range of [0.4, 1], and the total of $c_1 + c_2 \approx 4$.

3 A Computational Cognitive Model of Action Awareness

Action awareness is a complex cognitive phenomenon that covers the questions how action selection and execution contribute to the awareness and/or vice versa. Some evidence lead to a hypothesis that awareness of action selection is not directly causing the action execution (or behaviour) but comes afterward as an effect of unconscious processes of action preparation [17]–[21]. In contrast, another hypothesis claims that both predictive and inferential processes related to the action preparation and execution may contribute to the conscious awareness of the action [22]–[23]. By integrating these evidences with other necessary processes that contributes for awareness of action,

```
1. Initialize ω, c1, c2, n, m
2. For i=1 to n do
3.   Initialize each Xi with random values for variables
     (bounded by given max and min);
4.   Initialize each vi randomly;
5. End for
6. Initialize an array of tournaments' solutions (tSolutions)
   with size n
7. Do
8.   For each particle
9.     Calculate fitness value
10.    If the fitness value is better than its personal best (Pi)
11.      set current value as the new personal best
12.    End
13.    Choose the particle with the best fitness value in the
       swarm as current global best
14.    If the current global best value is better than its global
       best (Pg)
15       set current value as the new global best
16.    Else
17.      If convergence stays with the same Pg value for 'm'
         consecutive rounds
18.        add the Pg of current tournament into tSolutions
19.        If the no. of local solutions in lSolutions is n
20.          replace all current particle from particles in
             lSolutions and GOTO 8
21.        Else
22.          update each particle with random Xi and vi values
             and GOTO 8
23.        End
24.      End
25.    End
26.  For each particle
27.    Calculate particle velocity according to equation (1)
28.    Update particle position according to equation (2)
29.  End
30.  While maximum iterations (n) or minimum error
     criteria is not attained
```

Pseudocode 1: Pseudocode of this improved PSO

a computational cognitive model was developed that published in [24]. The same model is used in this paper for the proposed approach. Detailed information of cognitive, affective and behavioural neuroscience literature and a detail model description are not presented in this section but only an overview (for more details see [24]).

Fig. 1 presents the cognitive model developed for action awareness and Table 1 presents the abbreviations used in that. Model uses three world states (WS) as inputs: stimulus s_k, context c_k, and effect b_i. In addition, it includes two outputs to interact with the environment: EA(a_i) and EO(a_i, b_i, s_k, c_k). The input world states WS(s_k), and WS(c_k) lead to sensor states SS(s_k), and SS(c_k), and subsequently to sensory representation states SR(s_k), and SR(c_k), respectively. A sensory representation may trigger one or many action preparations: PA(a_i), in parallel and in unconscious (or automatic) form. The brain will evaluate the effect of each relevant action preparation by comparing the feelings: F(b_i), associated to each individual valuated effects (without actually executing them through the body loop) as proposed by Damasio [25], which is presented as "as-if body loop" (PA(a_i) → SR(b_i) → F(b_i)) in Fig. 1. The simulated option that has the strongest valuated feeling performs as a GO signal through the body loop and else are NO-GO options. Each PA(a_i) state suppresses its complementary options PA(a_j) for j≠i (as shown in dotted looped red arrow in Fig. 1) proportional to the accumulated strength of that option. This behaviour is in line with the explanation for the lateral inhibition in [26]. Therefore, naturally the strongest internally satisfied option (which is exceeding a threshold value) will become selected.

This model further includes action ownership states. Action ownership is a useful concept, which is mainly important to differentiate in how far a person attributes an action to him or herself, or to another person (see [27]). Model provides qualitative analysis on cognition before and after action execution (together with prior and retrospective awareness states). The ownership state includes information of inputs, action preparation, and predicted feeling of the preparing action option. This integrated information is used as the gateway to develop action awareness. Similar to ownership, awareness also separated into prior awareness and retrospective awareness as suggested by Haggard and co-workers [22], [23]. For each ownership state an associated awareness state may (or may not) emerge. Awareness states play a higher order cognitive role. The direct links from ownership and feeling to awareness state realise bottom-up activation (see [28], [29]). Conversely, the effects of awareness states on other states realise top-down activation, which is considered to be a conscious or

intended process (see [30]–[32]). Therefore in the presented model PAwr(a_i, b_i, s_k, c_k) is only affected by PO(a_i, b_i, s_k, c_k) and F(b_i). This is useful to model the idea of Benjamin Libet: brains initiate voluntary movements before we are aware of having decided to move [17]–[21]. Moreover PAwr(a_i, b_i, s_k, c_k) affects PA(a_i) and EA(a_i) and this is reflects the idea of Haggard and co-workers: there may be an impact from this subjective awareness state on action execution [22]–[23]. By this PAwr(a_i, b_i, s_k, c_k) to PA(a_i) link the agent can inject some bias to the current unconscious process through awareness. This may strengthen a weaker action option and improve the predictive feeling of that option (which may lead to getting it executed) [33]. Furthermore, in this model PAwr(a_i, b_i, s_k, c_k) can also directly strengthen the action execution state. By considering the proposed model as an agent, a particular behaviour trace is expected which is considered as main characteristics of the simulation example for parameter estimation. Expected trace is shared in an external appendix [1] (from [24]). The trace provides information to extract some important patterns to isolate the behaviour. The order of activation of the states is a very important feature. Furthermore, it is expected to be having a dip in the sensory representation and feeling in-between predictive and inferential representations [34]–[36]. In addition, the maximum strength of each state should exceed some threshold value to highlight the contribution of each state as the literature presented in [24].

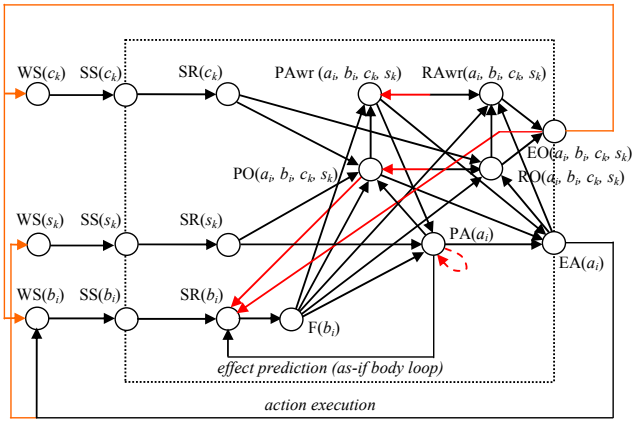

Figure 1: Overview of the computational cognitive agent model. Here an arrow ➜ represents a direct activation to state B from state A, an arrow ➜ represents a direct suppression to state B from state A, an arrow ⇢ represents a suppression to all the complements of 'ith' state on Bi from state Ai (where 'i' presents an instance of a particular state), and ➜ represents a direct supression to all parellel forms of that state. (from [27])

WS(W)	world state W (W can be either: context c_k, stimulus s_k, or effect b_i)
SS(W)	sensor state for W
SR(W)	sensory representation of W
PA(a_i)	preparation for action a_i
F(b_i)	feeling for action a_i after as-if loop or action execution
EA(a_i)	execution of action a_i
PO(a_i,b_i,c_k,s_k)	prior ownership state for action a_i with b_i, c_k, and s_k
RO(a_i,b_i,c_k,s_k)	retrospective ownership state for a_i with b_i, c_k, and s_k
PAwr(a_i,b_i,c_k,s_k)	prior-awareness state for action a_i with b_i, c_k, and s_k
RAwr(a_i,b_i,c_k,s_k)	retrospective-awareness state for action a_i with b_i, c_k, and s_k
EO(a_i,b_i,c_k,s_k)	communication of ownership and awareness of a_i with b_i, c_k, and s_k

Table 1: Nomenclature for Fig. 1

The proposed model is compiled to a computational model from a dynamic system perspective. The detail information of the model compilation and its parameter information (i.e., steepnesses: σ, thresholds: τ, weights: ω_k, update speed factors: γ, and step size: Δt) can be found in [24]. In our previous work for this, a systematic analytical driven parameter estimation method was used with realistic results (see [24]). That work includes 8 scenarios and only the first scenario will be considered for the scope of this paper. In [24] first scenario is focused on the interplay between conscious and unconscious processes on action selection where the prepared action has satisfactory predicted effects and therefore is executed; in this case both prior and retrospective awareness states occur. The behavioural results for first scenario in [24] is not presented in here. This result is used to compare the quality of proposed parameter estimation method through CS and PSO.

[1] http://www.few.vu.nl/~dte220/BICA2015_Appendix.pdf

4 Parameter Estimation with PSO

The model presented in Section 3 together with computational basis is considered for PSO based parameter estimation approach. The system includes 17 steepnesses, 17 thresholds, 39 weights, 2 update speed factors, and 1 step size. Each of these considered as variables and assigned domains as required in the CSP. Nevertheless, heuristic knowledge is used for assigning domain values for each variable to make the complexity less intractable and to prune the search space to increase the performance. A pre analytical analysis conducted to identify variables where its value can be pre determined. For example, WS(s_k), and WS(c_k) are the input states of the model and obviously σ values should be almost zero and therefore value 0.01 is used for these together with value 1.0 for τ. Furthermore, state F(b_i) affects only from SR(b_i) and therefore, it is obvious that from causality perspective F(b_i) should activate as soon as SR(b_i) has started to emerge. Therefore, σ value of SR(b_i) should also be almost zero and value 0.01 is used for this. In additionally, as SS(s_k), SS(c_k), SR(s_k), and SR(c_k) states includes direct representation of the inputs through weights ω_5, ω_6, ω_8, and ω_9 (see [24]); these assigned with value 1.0 (this is an implementation decision, if only a portion from input strength should be used for this then it can be < 1.0). Furthermore, according to the behaviour necessary for the highlighted scenario in the previous section, it is clear that weight values attached to some states should have very high values or above average values or very small values or negative values (high, average or small). Therefore, than using a domain range for weights 0 to 1 or 0 to -1, different ranges are used for small, average, and large (e.g., -0.7 to -0.4, -1.0/-0.9 to -0.6, 0.1 to 0.5, 0.4 to 0.7, 0.6 to 0.9/1.0). For steepness and threshold values also rather than using a generic range, custom ranges used with prior experience. From analytical perspective, in a model like this the states which are activates at the beginning normally shouldn't have very strong steepness and threshold values as they are directly coupled to the inputs and having strong weight values naturally more strength will be collected. Nevertheless states come later (especially after action execution) it is important to have very strong values for steepness and threshold variables. All value ranges used for variables available in an external file[2] (this includes the initial velocity values used in the PSO algorithm too). In addition to the variables and there domain values, a set of constraints (C1 to C5) also defined based on the neurocognitive evidences.

C1. State activation order should perceived: WS(s_k) & WS(c_k) → SS(s_k) & SS(c_k) → SR(s_k) & SR(c_k) → PA(a_i) → SR(b_i) → F(b_i) → PO(a_i, b_i, c_k, s_k) → PAwr(a_i, b_i, c_k, s_k) → EA(a_i) → WS(b_i) → SS(b_i) → RO(a_i, b_i, c_k, s_k) → RAwr(a_i, b_i, c_k, s_k) → EO(a_i, b_i, c_k, s_k)

C2. Each state value should converge to zero after retrospective effects

C3. All states except SR(b_i) and F(b_i) should only provide one peak behaviour

C4. States SR(b_i) and F(b_i) should provide two peak behaviour where the value at first peeks should be considerably small respective to the value at the second peak

C5. Peak value (max) of each state should be equal or exceed a given value: 1, 1, 0.55, 0.55, 0.50, 0.50, 0.65, 0.50, 0.50, 0.65, 0.65, 0.85, 0.70, 0.60, 0.65, 0.65, 0.70 (hear the order is the same order of states in C1)

These constraints are noted as the most influential restrictions to have necessary generic behaviour. There is a trade off between minimum and maximum constraints required for a model to isolate a generic parameter value set to mimic its behaviour with reality (see [8]). Each constraint associated with a negative error value such that if all constraints are satisfied the value of the error is zero. For the constraint C1, -13 is assigned as the total error and if two consecutive positions are as in the order then +1 will be assigned (therefore, if all the states are in the expected order the error is 0). For the constraint C2, -17 is assigned as the total error and if a selected state is converging to an activation strength of 0 after the retrospective effect then it will get +1. For C3 and C5 also the same above rule applied for each state. For the constraint C4, -2 is assigned as the total error and if a selected state is showing a two peak behaviour it gets +1. Having represented the proposed model as a CSP, it is

[2] http://www.few.vu.nl/~dte220/IAT15_ModelVariablesS1.xml

solved by using the proposed PSO algorithm. For this purpose, following configuration parameters are used for the PSO equations (1 & 2) presented in Section 2: number of particles (n) is 30, weight of inertia for the velocity (ω) is 1, local acceleration coefficient (c_1) is 2, global acceleration coefficient (c_2) is 2. These selected parameter values are according to the guidelines given in [12], [15]. Having these parameters for the PSO equations (1 & 2) the model is implemented in the Java language and passed the initial values in a XML data file[2]. The pseudocode presented in the Section 2.2 is used for the PSO implementation. First 30 particles were randomly initialized for the 73 variables (17 steepnesses, 17 thresholds, 39 weights) under the domain ranges provided for each variable. For the low and fast update speed factors (γ) values 0.6 and 0.7 are used respectively together with value 0.25 as the step size (Δt). Having the same value for local acceleration coefficient (c1) and global acceleration coefficient (c2) the swarm is not biased to local or global exploration but due to the random values assigned for r_1 and r_2 (see equations 1 & 2) in each iteration swarm may randomly select a local exploration or global exploration.

Having 30 random particles in the search space, they evaluate the quality of current positions individually by mapping with the five constraints. If the current local best position is better than the past best local position visited (only after the first iteration) it will be replaced by the current. Having identified the quality of current positions by each particle, they share the information with each other to find the current global best position. If the current global best position is better than the previous global best position (only after the first iteration) as a swarm, then all particles change their global best to the current global best. In additionally, each particle changes its current position towards to the resultant of global best position (i.e., social aspect), local best solution (i.e., cognitive aspect), and speed factor. This process executes as a tournament. In each tournament, if any particle's position able to satisfy all the constraints, then the process will be terminated and the variable values of that particle's position will be used as the parameter values in the model. If a tournament fails to find a particle that satisfies all the constraints then the global best position of the swarm at the end of that tournament will be separately saved and new tournament will be initiated with random positions for all particles. It is decided that a tournament is failed if the same error value for constraints holds for 15 iterations continuously. After a 30 tournaments if the system is unable to find a solution the captured best global positions in each tournament will be used and create a swarm of 30 particles and let the system to converge. In this case, there is a very high probability to have many particles with the same low error values (most probably very closer to the 0) and therefore, speed of particle movements will be limited but more local explorations performs. This particular improvement introduced as a solution for premature convergence problem. A major drawback in PSO is its premature convergence ([15], [16]) and putting quality particles after 30 tournaments improve the convergence speed a lot and due to this elicit selection feature, the system will not discard better solutions for future explorations. This iterative process continues until a particle is obtained a position that satisfies all the constraints. Fig 2 shows the behaviour patterns obtained from the proposed PSO algorithm for the mentioned cognitive model. All the results are completely satisfied with all the constraints and align with the evaluation in [24]. There are three different solutions included in the Fig 2, which are provided by three different executions. Nevertheless, all the results are fully satisfied by the expectations (constraints).

5 Discussion

Parameter estimation is a challenging task in many application domains, including the dynamic cognitive modelling domain. Nevertheless, it is difficult to adapt common parameter estimation algorithms for cognitive models where usually there are only limited empirical data available (a.o. due to limitations in neuro-imaging techniques and the complexity of the human brain). Moreover, most of the behaviours are presented in the form of features or patterns over time that are explained in non-quantitative and discrete forms.

This paper presents a solution for such cognitive models using an improved PSO algorithm. The complex cognitive model used in this paper has been published in [24] together with eight scenarios to validate the model behaviour. Only the first simulation of that work was considered in this paper as an illustration for parameter estimation, and acceptable results were achieved. According to these results it is confirmed that this new approach is suitable for parameter estimation for these types of complex cognitive models. Initially for this purpose a standard parameter estimation approach was used without introducing a tournament based improvement. It was found that then the method is unable to converge to a solution and get tapped with local minima all the time. When exploring approaches to eliminate premature convergence issues, there are some techniques available (see [16]). Nevertheless, all of these techniques are having limitations and there are some technical problems to adapt those techniques to the cognitive domain.

The Multi-Swarm PSO (MSPSO) technique ([37], [38]) was chosen as a suitable solution for this issue and it has many supporting features for parameter

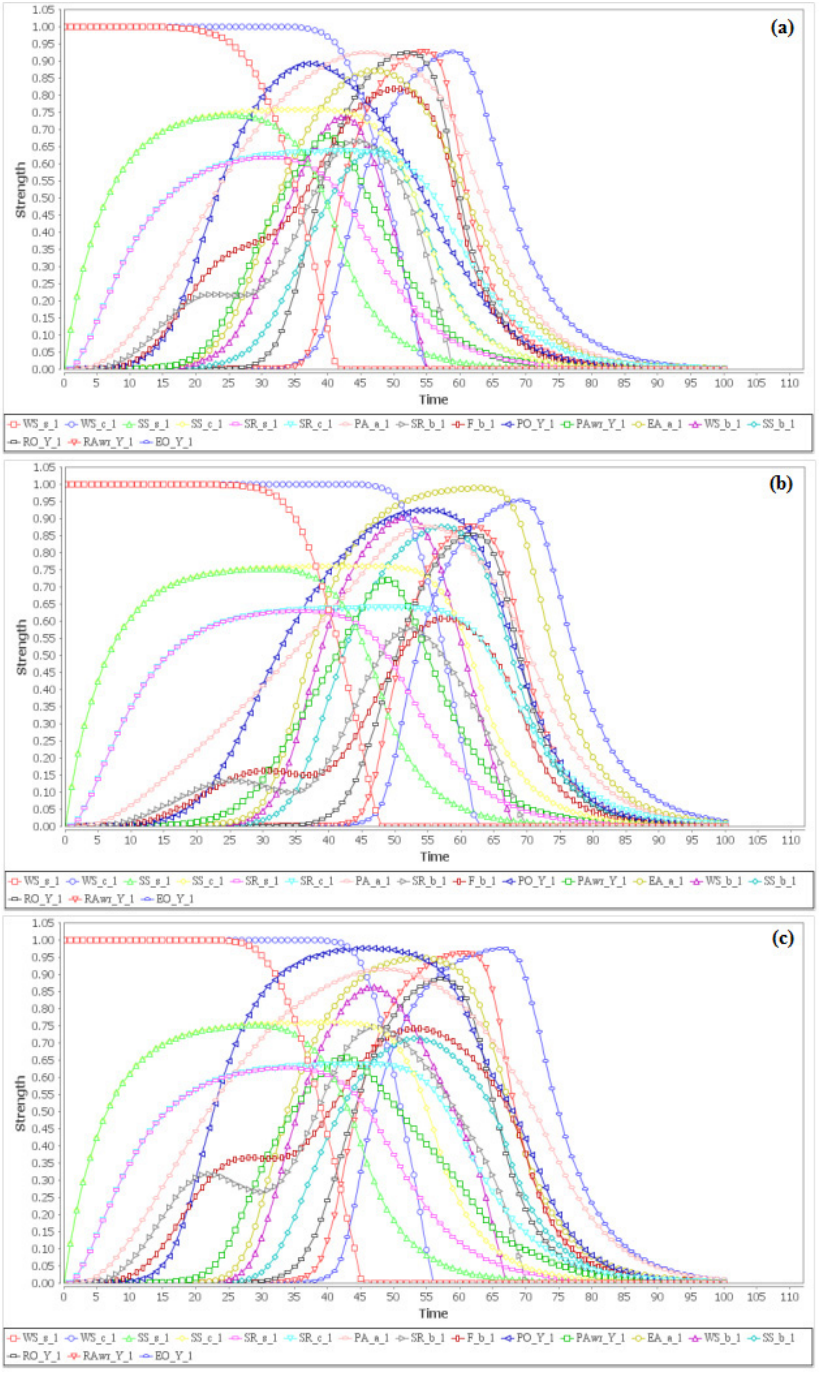

Fig. 2. Cognitive behaviour obtained through PSO baded paramter estimation algorithm for executing an action with ownership and awareness

estimation in dynamic cognitive models. The most common approach in the MSPSO is to partition the current swarm into several sub-swarms. Nevertheless, due to computational problems in parallel searching and difficulties in forming sub-swarms as equally representing the partitions of the search space, this is also not providing promising results. Therefore, in this paper a new approach is presented by introducing a tournaments based approach. Therefore, no parallel searching is required and having a strong random particle generator it will guarantee a quality distribution of particles in the search space. It is a future work to conduct a systematic comparison this with MSPSO. According to the results in Fig 2 it is clear that this approach is not just giving one solution but multiple near optimal solutions. This seems like a problem as it is not able to properly represent a generic behaviour of the cognitive model. This problem is addressed in our previous work [24] by introducing 8 different scenarios (which are interrelated from a functional point of view) and finding a unique parameter value set that captures the behaviour of all the simulations. Having interrelated scenarios, the goal is to find a global parameter value set that satisfies all the of those individually (for this it is necessary to have many near optimal solutions at the beginning, but later the system will converge to a one solution due to the feature of interrelatedness among scenarios). This is a future work for this technique to combining the procedure mentioned in [24] for this PSO algorithm. In additionally more validations are required for this approach and it is essential to do more thorough analysis on effects of changing number of particles, changing the parameter values of the PSO algorithm, removing/diluting heuristic knowledge used to set domain ranges, and measuring average running time. Also, having thoughts to present this cognitive model as a generic framework to use as an experimental workbench on action selection related cognitive phenomenon, it is important to have a feature that the modeler does not need to do any programming to obtain the behaviour. The current implementation only needs an input in a XML file and results will be generated except representing the constraints in Java. Having a formal representation to present the constraints will eliminate this limitation and a predicate logic based representation will be used for this in future work.

Acknowledgment

I wish to thank Prof. Jan Treur at VU University Amsterdam, for his great support and supervision in all the phases of this work.

References

1. Bandettini PA, “What’s New in Neuroimaging Methods?,” Ann. N. Y. Acad. Sci.; 2009, 1156(1), p. 260–293.
2. Busemeyer JR, Diederich A. Cognitive modeling. Los Angeles: Sage, 2010.
3. Shiffrin RM. “Perspectives on Modeling in Cognitive Science,” Top. Cogn. Sci. 2010:2 (4):736–750.
4. McClelland JL. “The Place of Modeling in Cognitive Science,” Top. Cogn. Sci. 2009:1 (1):11–38.
5. Addyman C, French RM. “Computational Modeling in Cognitive Science: A Manifesto for Change,” Top. Cogn. Sci. 2012:4 (3):332–341.
6. Lewandowsky S, Farrell S. Computational modeling in cognition: principles and practice. Thousand Oaks: Sage Publications, 2011.
7. van den Bos A. Parameter estimation for scientists and engineers. Hoboken, N.J: Wiley-Interscience, 2007.
8. Kumar V. “Algorithms for Constraint-Satisfaction Problems: A survey,” AI Mag. 1992:13 (1), 32–44.
9. Tsang E. Foundations of constraint satisfaction. London; San Diego: Academic Press, 1993.

10. van Rooij I. “The Tractable Cognition Thesis,” Cogn. Sci. Multidiscip. J.; 2008:32 (6), p. 939–984.
11. Bulatov A, Jeavons P, Krokhin A. “Classifying the Complexity of Constraints Using Finite Algebras,” SIAM J. Comput.; 2005:34 (3), p. 720–742.
12. Bai Q. “Analysis of Particle Swarm Optimization Algorithm,” Comput. Inf. Sci.; 2010:3 (1), p. 180–184.
13. Kennedy J, Eberhart R. “Particle swarm optimization,”; 1995:4, p. 1942–1948.
14. Rini DP, Shamsuddin SM, Yuhaniz SS. “Particle Swarm Optimization: Technique, System and Challenges,” Int. J. Comput. Appl.; 2011:14 (1), p. 19–27.
15. Poli R, Kennedy J, Blackwell T., “Particle swarm optimization: An overview,” Swarm Intell.; 2007:1 (1), p. 33–57.
16. Nakisa B, Nazri MZA, Rastgoo MN, Abdullah S. “A survey: Particle swarm optimization based algorithms to solve premature convergence problem.,” J. Comput. Sci.; 2014: 10 (9), p. 1758–1765.
17. D’Ostilio K, Garraux G. “Brain mechanisms underlying automatic and unconscious control of motor action,” Front. Hum. Neurosci.; 2012:6.
18. Haynes JD. “Decoding and predicting intentions: Predicting intentions,” Ann. N. Y. Acad. Sci.; 2011:1224 (1), p. 9–21.
19. Baumeister RF, Masicampo EJ, Vohs KD. “Do Conscious Thoughts Cause Behavior?,” Annu. Rev. Psychol.; 2011:62 (1), p. 331–361.
20. Wegner DM. The illusion of conscious will. Cambridge, Mass: MIT Press, 2002.
21. Libet B, Gleason CA, Wright EW, Pearl DK. “Time of conscious intention to act in relation to onset of cerebral activity (readiness-potential). The unconscious initiation of a freely voluntary act.,” Brain; 1983: 106 (3), p. 623–642.
22. Haggard P, Clark S, Kalogeras J. “Voluntary action and conscious awareness,” Nat. Neurosci.; 2002:5 (4), p. 382–385.
23. Walsh E, Haggard P. “Action, prediction, and temporal awareness,” Acta Psychol. (Amst.); 2013:142 (2), p. 220–229.
24. Thilakarathne DJ, Treur J. “Computational Cognitive Modelling of Action Awareness: Prior and Retrospective,” Brain Inform.; 2015, p. 01–30.
25. Damasio AR. Self comes to mind: constructing the conscious brain. New York: Vintage Books, 2012.
26. Aron AR. “The Neural Basis of Inhibition in Cognitive Control,” The Neuroscientist; 2007:13 (3), p. 214–228.
27. Treur J. “A computational agent model incorporating prior and retrospective ownership states for actions,” Biol. Inspired Cogn. Archit.; 2012:2, p. 54–67.
28. Miller EK, Cohen JD. “An integrative theory of prefrontal cortex function,” Annu. Rev. Neurosci.; 2001:24 (1), p. 167–202.
29. Katsuki F, Constantinidis C, “Bottom-Up and Top-Down Attention: Different Processes and Overlapping Neural Systems,” The Neuroscientist; 2014:20 (5), p. 509–521.
30. Moore J, Haggard P. “Awareness of action: Inference and prediction,” Conscious. Cogn.; 2008:17 (1), p. 136–144.
31. Engel AK, Fries P, Singer W. “Dynamic predictions: Oscillations and synchrony in top–down processing,” Nat. Rev. Neurosci.; 2001:2 (10), p. 704–716.
32. Baluch F, Itti L. “Mechanisms of top-down attention,” Trends Neurosci.; 2011:34 (4), p. 210–224.
33. Kiefer M. “Top-down modulation of unconscious ‘automatic’ processes: A gating framework.,” Adv. Cogn. Psychol.; 2007:3, p. 289–306.
34. Blakemore SJ, Wolpert D, Frith C. “Why can’t you tickle yourself?,” NeuroReport; 2000:11 (11), p. R11–R16.
35. Blakemore SJ, Frith CD, and Wolpert DM, “Spatio-Temporal Prediction Modulates the Perception of Self-Produced Stimuli,” J. Cogn. Neurosci.; 1999:11 (5), p. 551–559.
36. Fourneret P, de Vignemont F, Franck N, Slachevsky A, Dubois B, Jeannerod M. “Perception of self-generated action in schizophrenia,” Cognit. Neuropsychiatry; 2002:7 (2), p. 139–156.
37. Li J, Xiao X. “Multi-Swarm and Multi-Best particle swarm optimization algorithm,” 2008, p. 6281–6286.
38. Zhao SZ, Suganthan PN, Das S. “Dynamic multi-swarm particle swarm optimizer with sub-regional harmony search,” 2010, p. 1–8.